Stroke filtering, dynamics & processing

Karan Singh

dgp
Dynamic Graphics Project
University of Toronto
www.dgp.toronto.edu

Issues in digital sketching

2D

- stroke filtering — fairing, clothoids…
- stroke processing — segmentation, recognition, regularization…
- stroke appearance — NPR, stylization…
- stroke dynamics — pressure, tilt, direction, temporal order…
- seamless UI Control — widgets, gestures, crossing, multi-stroke…
- navigation — paper manip., onion skinning…
- 2D curve creation — What are desirable curves, how do we perceive them in relation to our design knowledge?
- stroke perception — What spatio-temporal information do they convey?

3D (Additional dimension for 3D design, animation or 2D design explorations)

- 3D Navigation (camera tools, single/multi-view, view bookmarks…)
- 3D curve creation (2D stroke to 3D curves perception & inference…)
- animation (motion trails, evolving shape fronts…)
- alternate Designs (co-locating them in space…)

Stroke filtering: noise & error sources

- User error
  - Intent (wants a square but draws a rectangle).
  - Execution (unsteady hand).
  - Ergonomic (awkard drawing posture).
- Device error
  - Input (tablets better than mice or trackpads).
  - Resolution (projected better than surface capacitance).
  - Signal Noise.

What are desirable strokes?

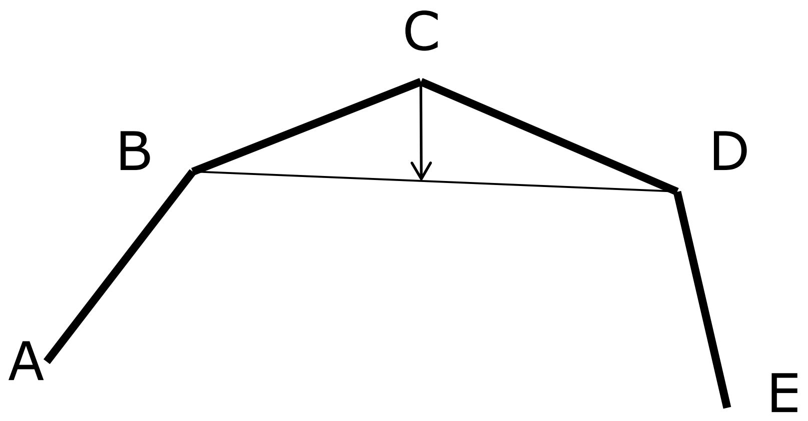
- **Smoothness**: *"tangent and perhaps curvature continuous curves"* [Farin et al. 87].

Simple smoothing approaches

- Laplacian. (neighbour averaging)
- Bi-Laplacian.
- LSQ spline fitting.

Simple smoothing approaches

- Laplacian. (neighbour averaging)
- Bi-Laplacian.
- LSQ spline fitting.



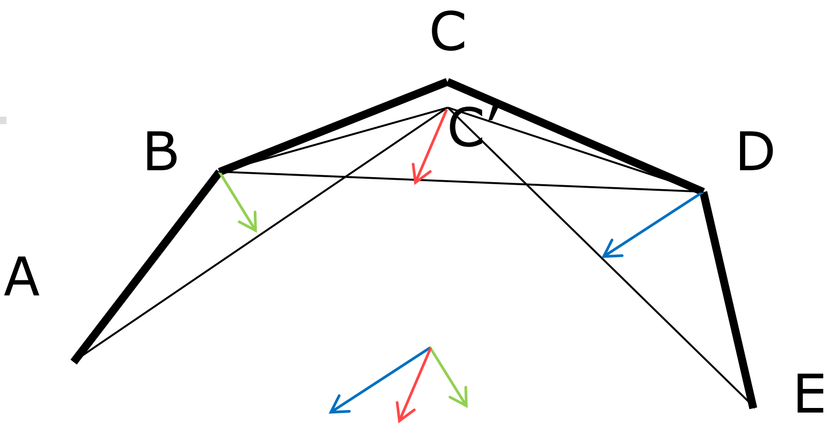
$lap(C) = (B+D)/2 - C$

$C' = C + d * lap(C) \qquad 0 < d < 1$

Best to run many iterations with
A small d, for eg. 5 iterations d=0.2

Simple smoothing approaches

- Laplacian. (neighbour averaging)
- Bi-Laplacian.
- LSQ spline fitting.



Find a C′ such that:

lap(C′) = (*lap*(B) + *lap*(D))/2

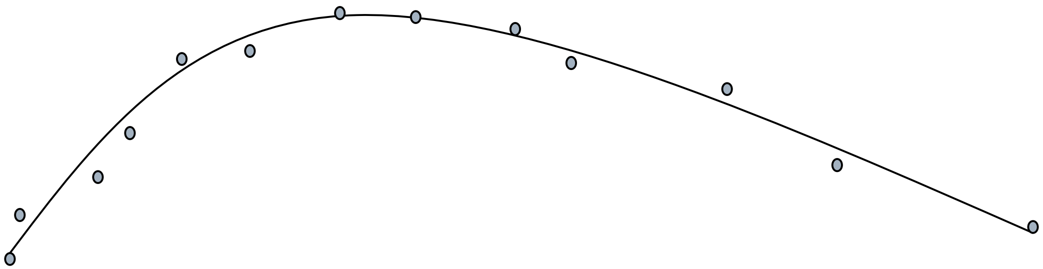
(B+D)/2-C′= (((A+C′)/2-B)+((E+C′)/2-D))/2

C′= 2/3 (B+D-A/4-E/4)

bi-lap(C)=C′-C

Simple smoothing approaches

- Laplacian. (neighbour averaging)
- Bi-Laplacian.
- LSQ spline fitting.

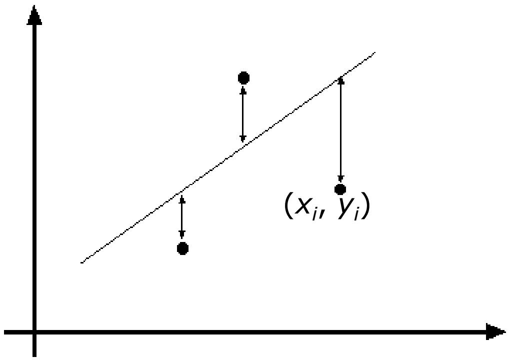


$f(t)=(x,y)$ from points (x_i, y_i)
Non-linear problem: guess t_i, LSQ, refine t_i, iterate…

LSQ line fitting

Data: $(x_1, y_1), \ldots, (x_n, y_n)$ ➡ Line equation: $y_i = m\, x_i + b$

Find (m, b) to minimize: $\boxed{E = \sum_{i=1}^{n} (y_i - m x_i - b)^2}$

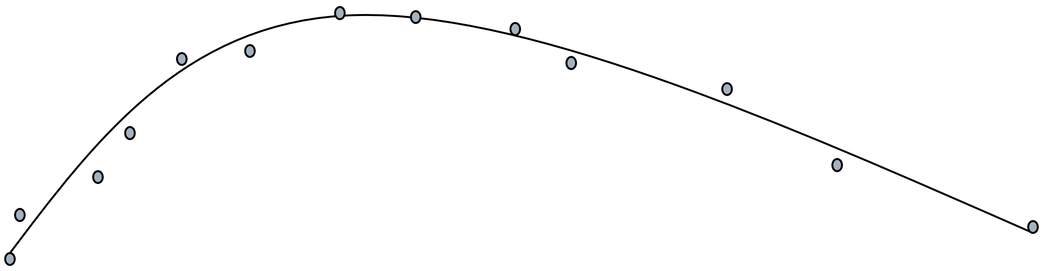


$$E = \sum_{i=1}^{n} \left(\begin{bmatrix} x_i & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - y_i \right)^2 = \left\| \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\|^2 = \|\mathbf{Ap} - \mathbf{y}\|^2 = \mathbf{y}^T \mathbf{y} - 2(\mathbf{Ap})^T \mathbf{y} + (\mathbf{Ap})^T (\mathbf{Ap})$$

$$\frac{dE}{dP} = 2\mathbf{A}^T \mathbf{Ap} - 2\mathbf{A}^T \mathbf{y} = 0 \quad ➡ \quad \mathbf{A}^T \mathbf{Ap} = \mathbf{A}^T \mathbf{y} \Rightarrow \mathbf{p} = \left(\mathbf{A}^T \mathbf{A}\right)^{-1} \mathbf{A}^T \mathbf{y}$$

Simple smoothing approaches

- Laplacian. (neighbour averaging)
- Bi-Laplacian.
- LSQ spline fitting.

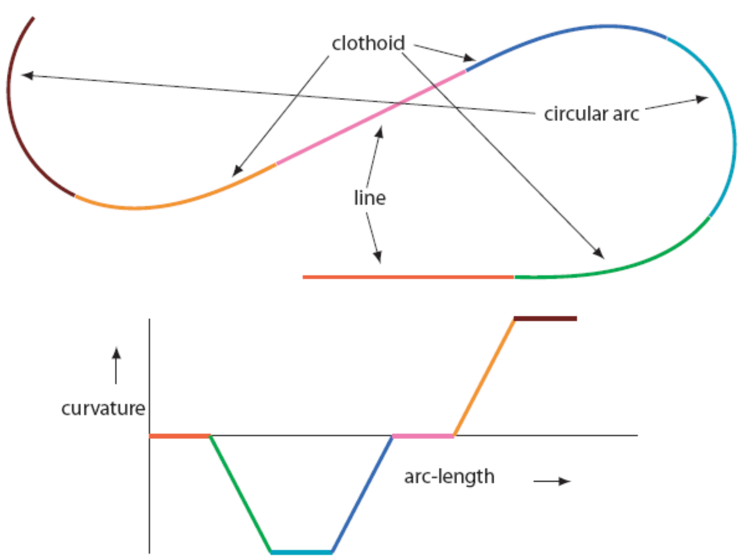


LSQ error minimizes f such that [$\text{sum}_i \; || f(t_i)-(x_i, y_i) ||^2$]
guess t_i, LSQ, refine t_i, iterate…

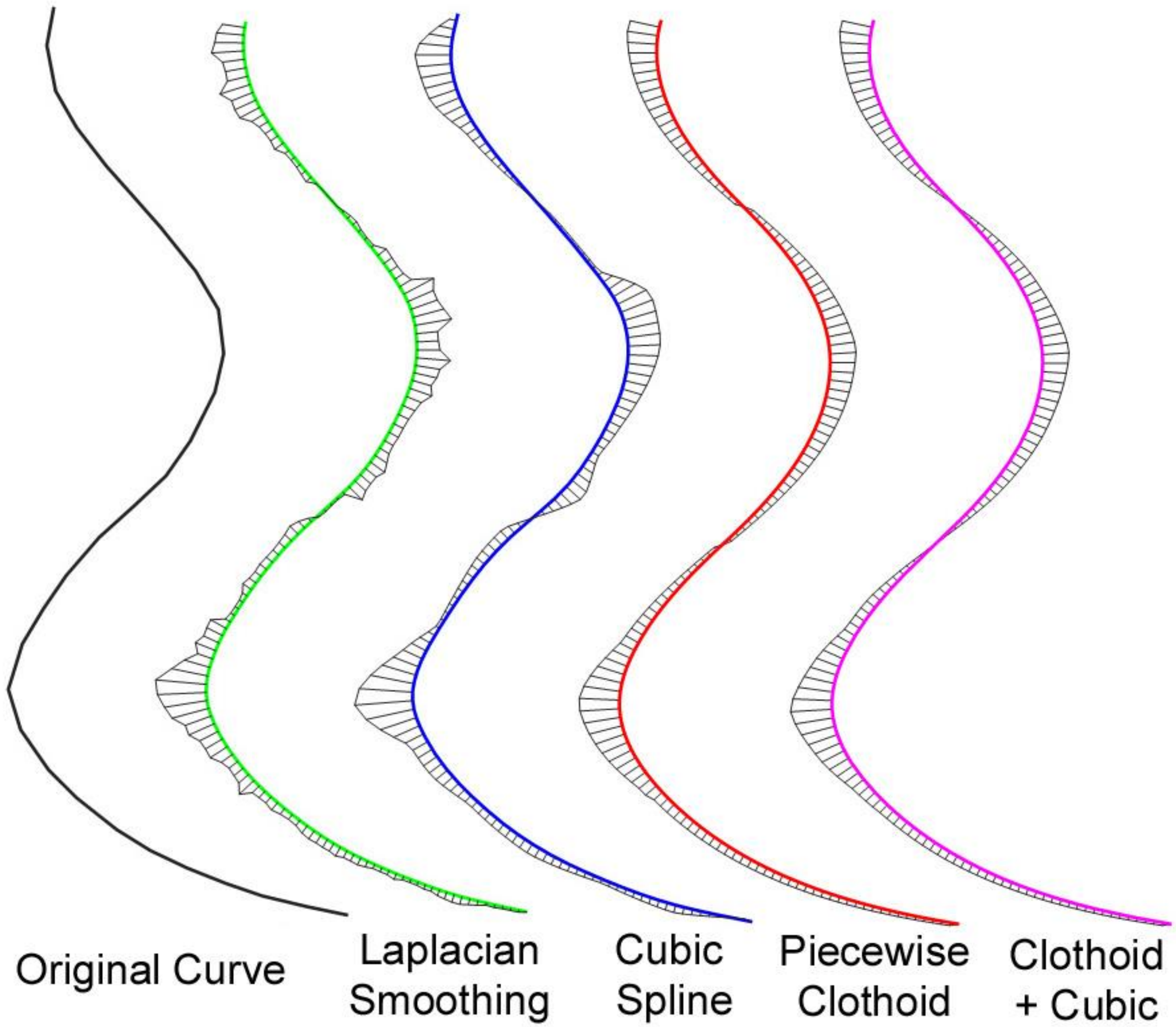
What are desirable strokes?

- **Fairness**: "*curvature continuous curves with a small number of segments of almost piecewise linear curvature*" [Farin et al. 87].

- Lines, circles and clothoids are the simplest primitives in curvature space.



Comparative approaches to fairing



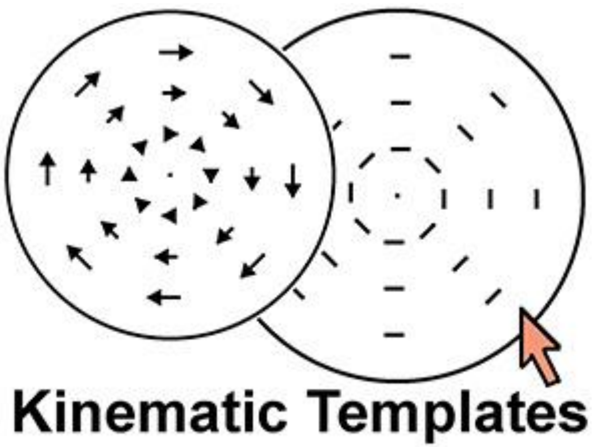
Stroke filtering

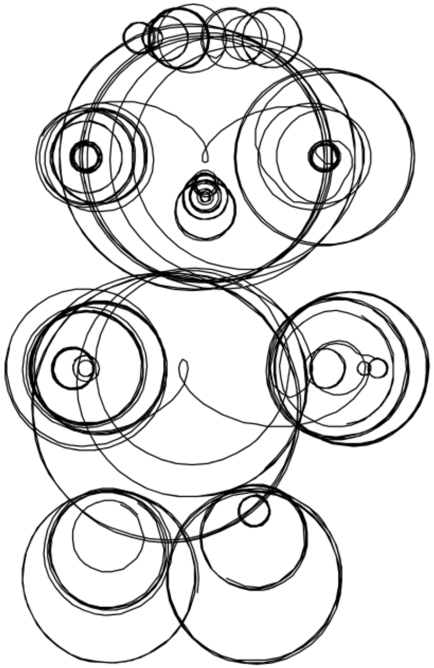
- **Neatness**: *"a combination of fairness and fine detail as intended by the user".*

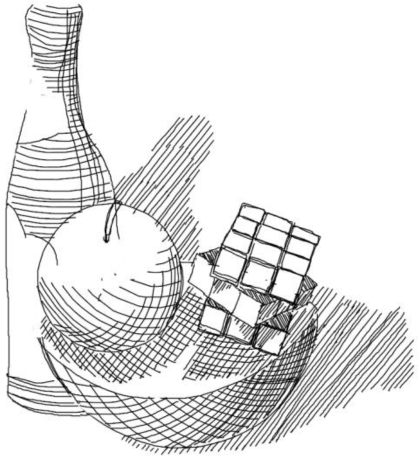
- Requires either implicit knowledge of user-intent, or an explicit neatening directive by the user.

Stroke filtering: Kinematic templates

Filter cursor position to lie along proximal flow-lines of pre-defined templates. (UIST 2008)







Stroke neatening: French curves

- Physical tools, used to model curves.

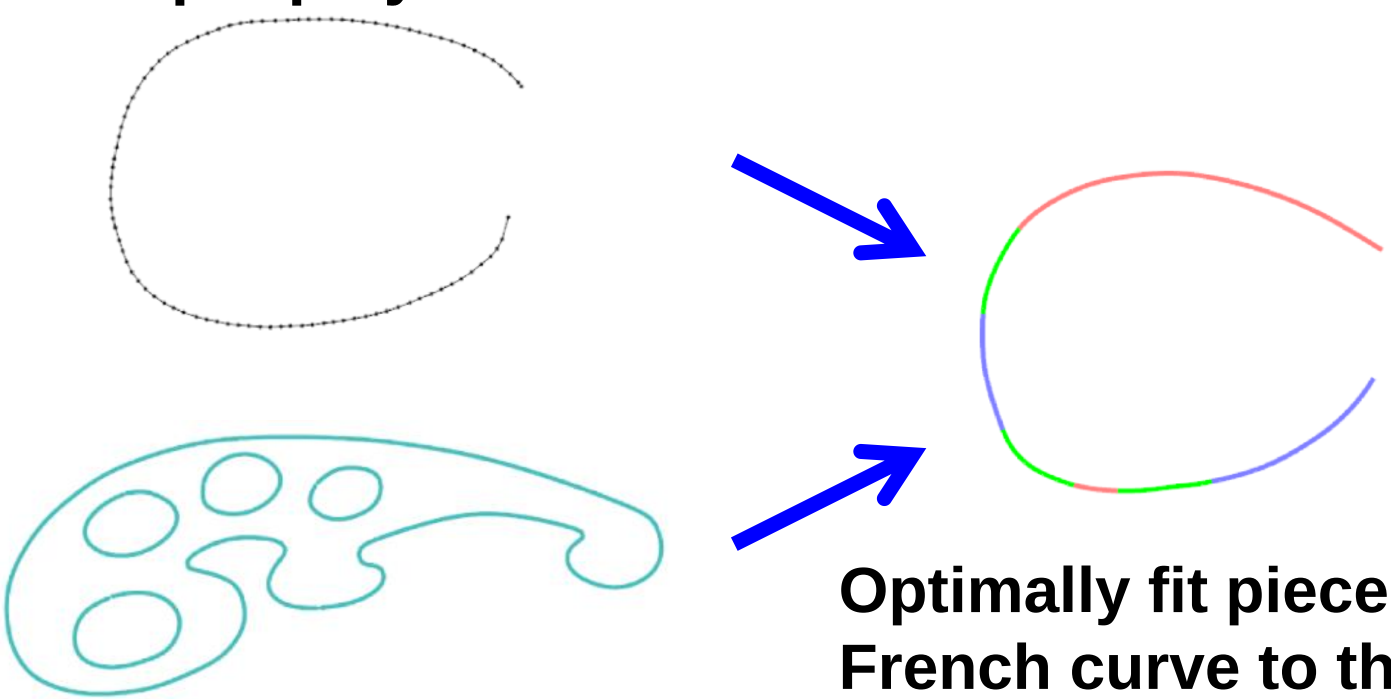
French curves +sketch interface

smooth shape priors,
specify a style/standard

fluid free-form

Stroke neatening

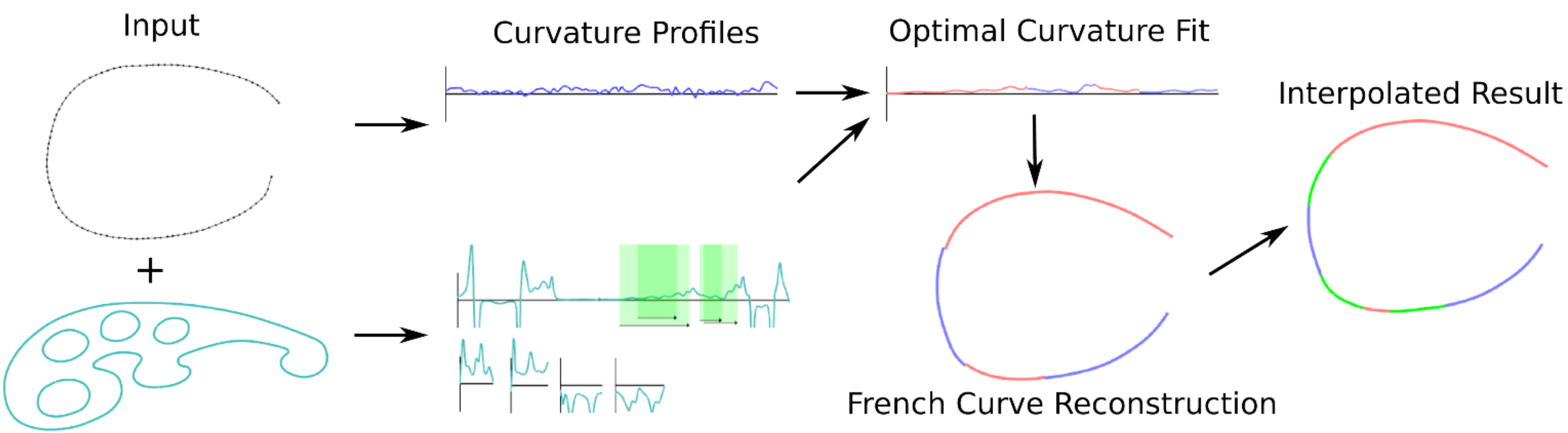
input polyline



French curve

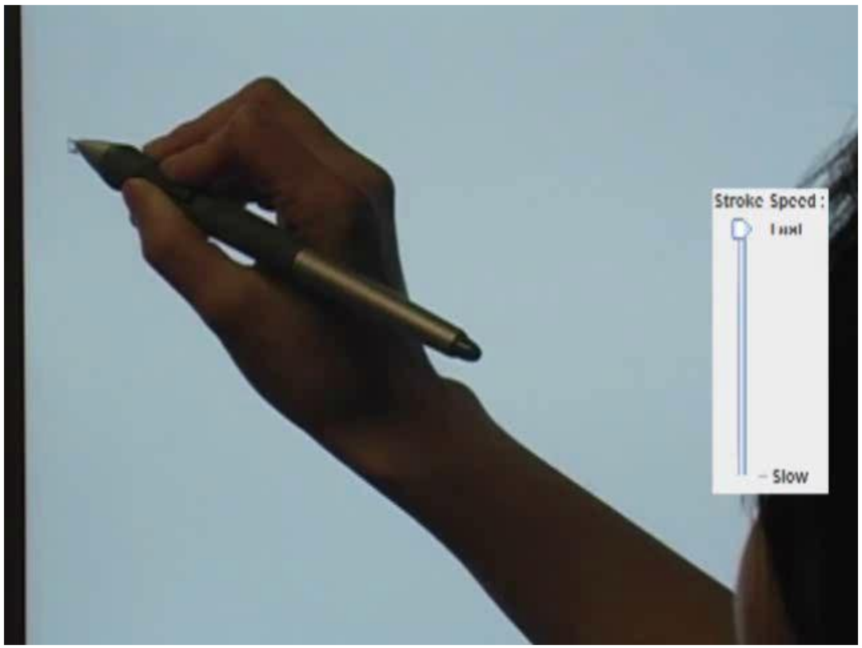
Optimally fit pieces of the French curve to the input

Approach



Input
+
Curvature Profiles
Optimal Curvature Fit
Interpolated Result
French Curve Reconstruction

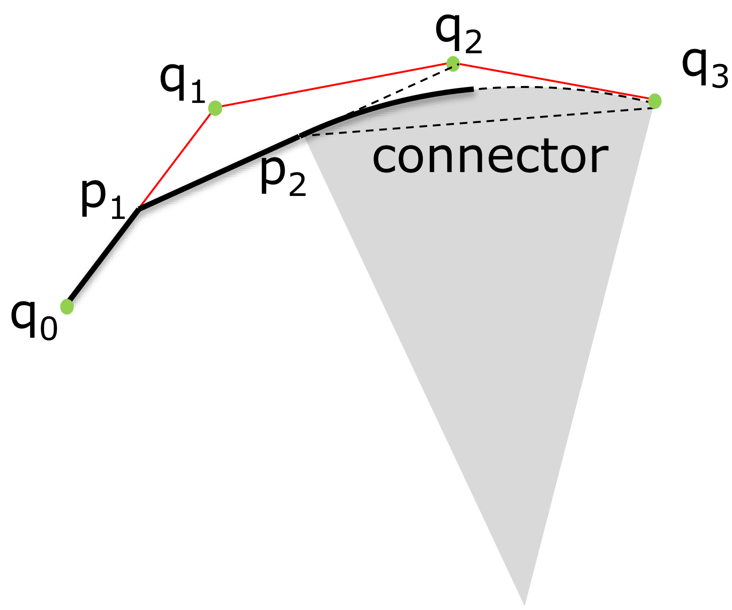
Stroke neatening & dynamics: elasticurves



[**Thiel, Singh**, **Balakrishnan** Elasticurves: Exploiting Stroke Dynamics and Inertia for the Real-time Neatening of Sketched 2D Curves, *UIST 2011*]
http://www.dgp.toronto.edu/~ythiel/Elasticurves/

Elasticurve

Input q_i's sampled at a time interval of *dt*

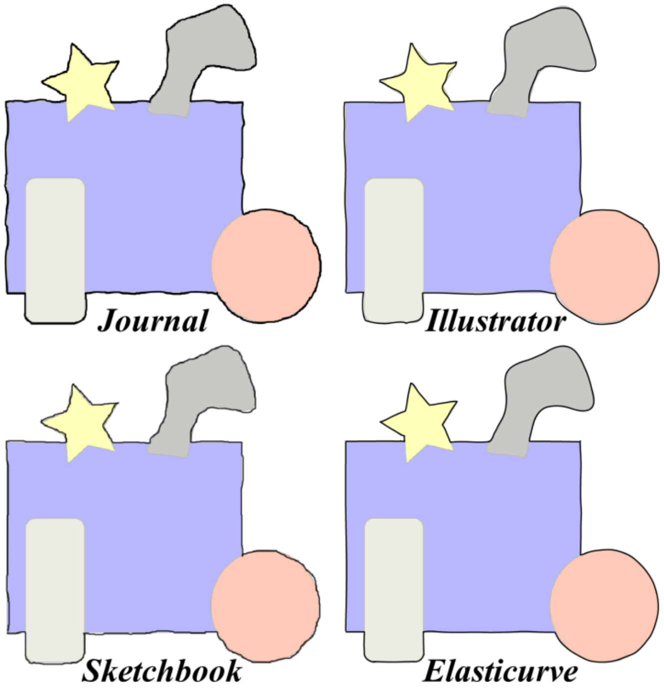


responsiveness = connector arc-length fraction extending an elasticurve.

Elasticurve Properties

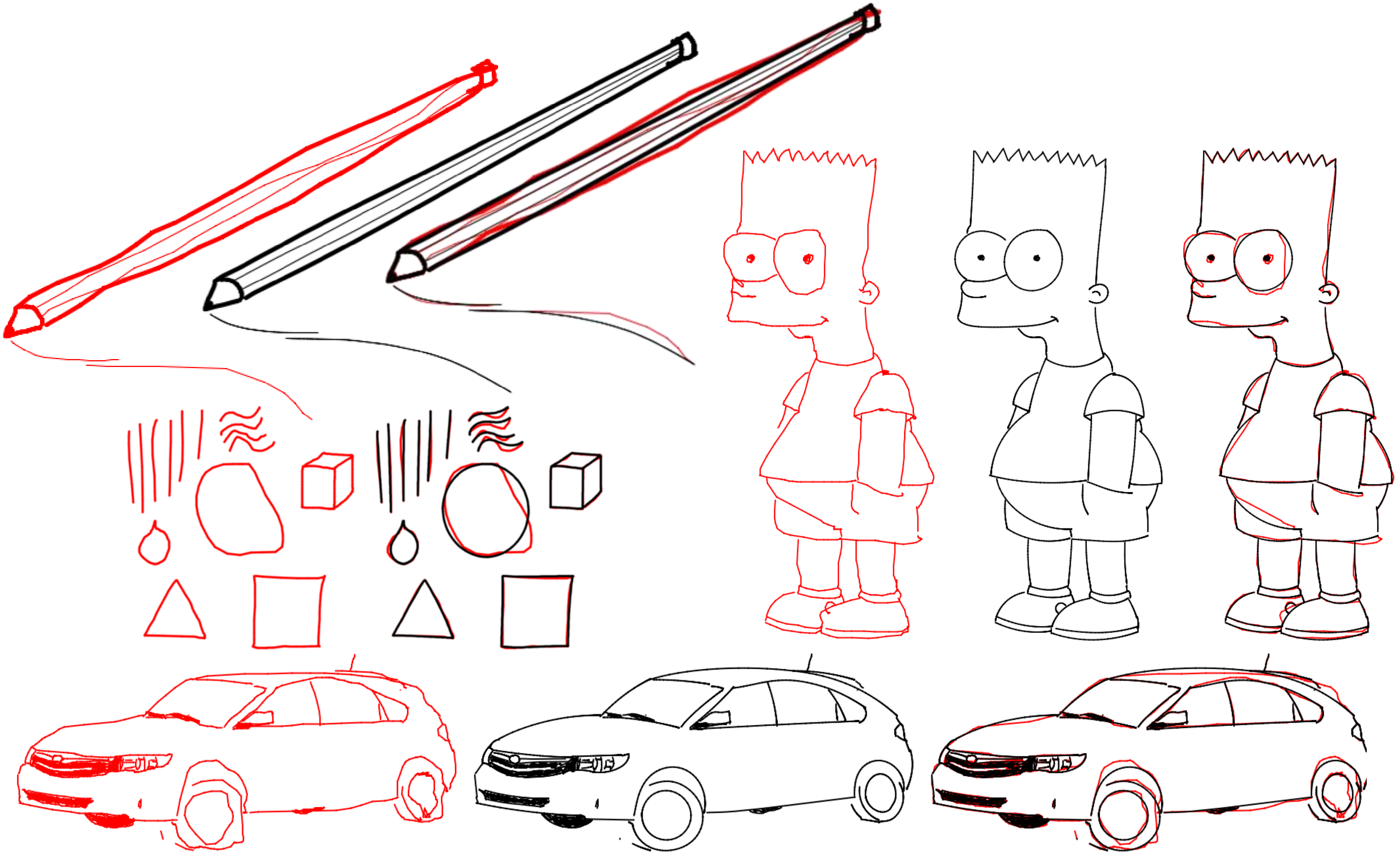
- **Explicit and real-time**: neatness is directly correlated to drawing speed and *responsiveness*.
- **Analytic:** resilience to *dt* sampling variation.
- **Precise:** embodies desirable shapes as connectors.

Elasticurve evaluation & curve quality



Intermediate user, trackpad,
visual best of 7 attempts.

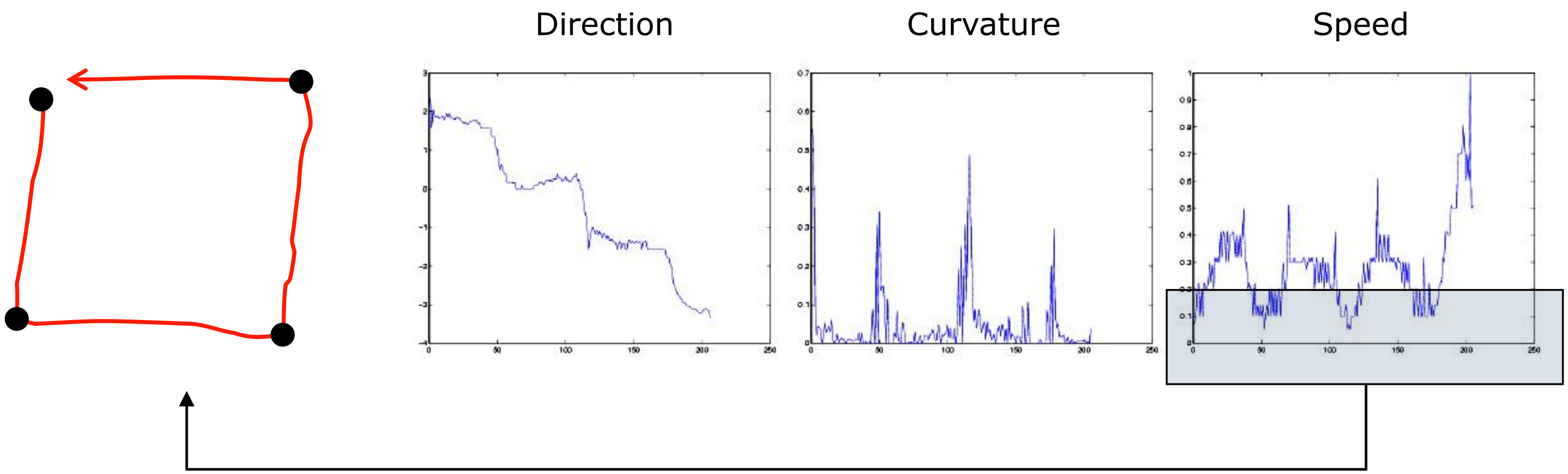
Elasticurve Results: novice/mouse



Stroke Processing

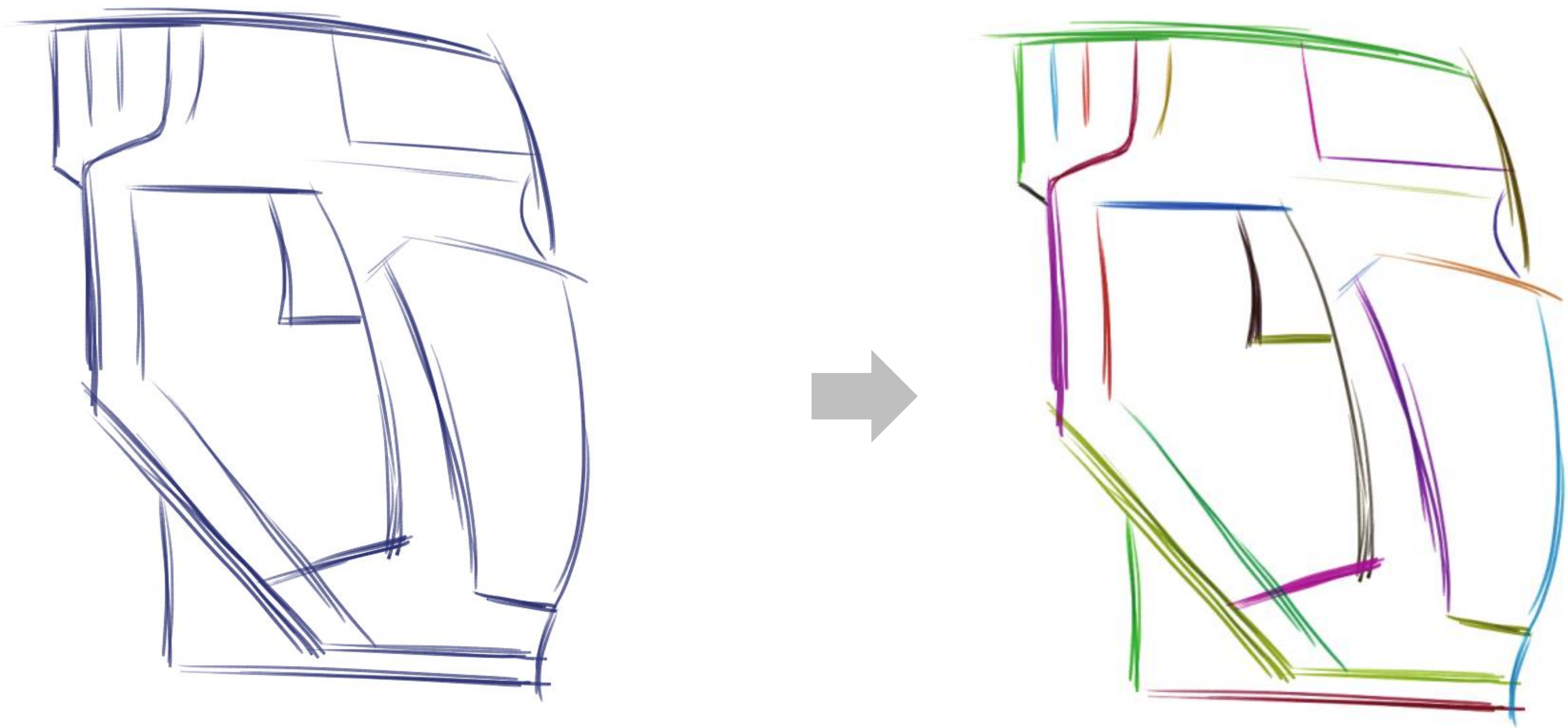
- *Filtering, neatening, beautification* can also be considered as stroke processing.
- Segmentation, classification, recognition.
- Regularization.
- Abstraction.
- Oversketching.
- Gestures.

Stroke segmentation: finding corners

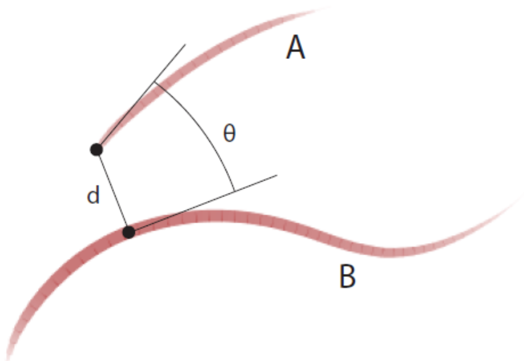


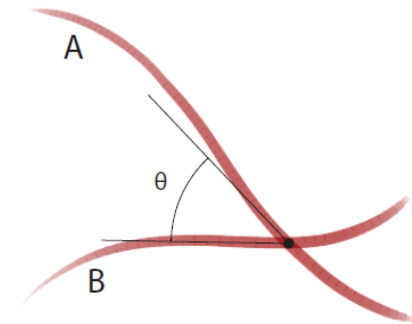
[**T. Sezgin et al.**, *Sketch Based Interfaces: Early Processing for Sketch Understanding*, Workshop on Perceptive User Interfaces, 2001.]

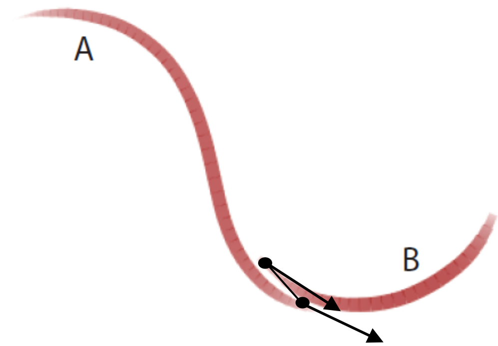
Stroke classification: pentamenti



Geometric Stroke Features







Proximity

$$d_{AB} = \|\mathbf{x}_i - \mathbf{x}_j\|$$

Alignment

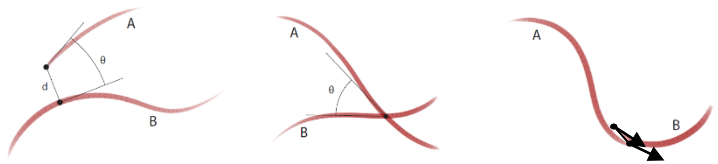
$$a_{AB} = \frac{|\angle(\mathbf{n}_A, \mathbf{n}_B)|}{\pi/2}$$

Continuity

$$c_{AB} = \frac{\|(\mathbf{n}_A \times \mathbf{n}_B)\| + \|(\mathbf{n}_A \times \mathbf{n}_s)\| + \|(\mathbf{n}_B \times \mathbf{n}_s)\|}{3}|s|$$

- Pairwise features
- Stroke proximity
- Local learning

Group Strokes by Affinity

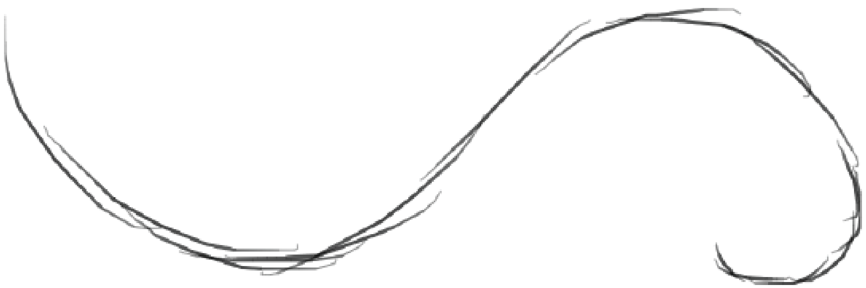


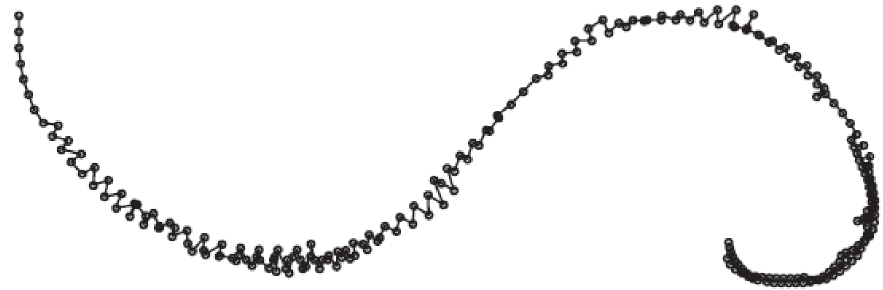
Affinity = Proximity + Alignment + Continuity

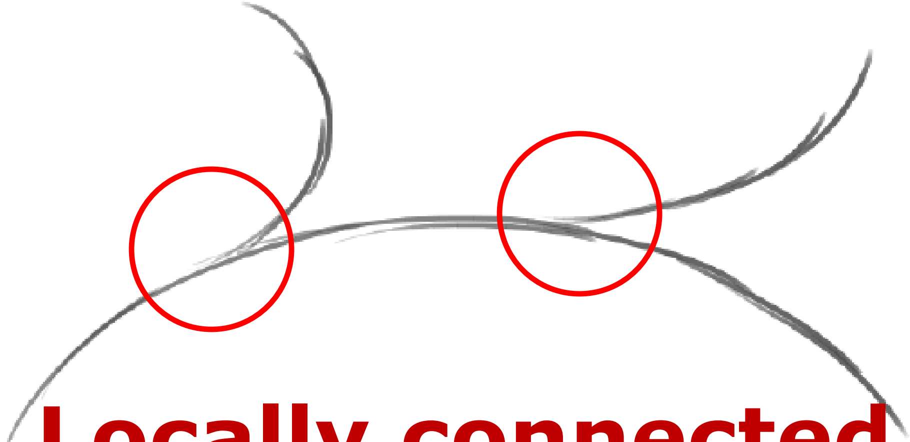
learning approaches with or without examples:

- neural network
- spectral clustering
- greedy grouping (single-link clustering)

Order stroke points parametrically





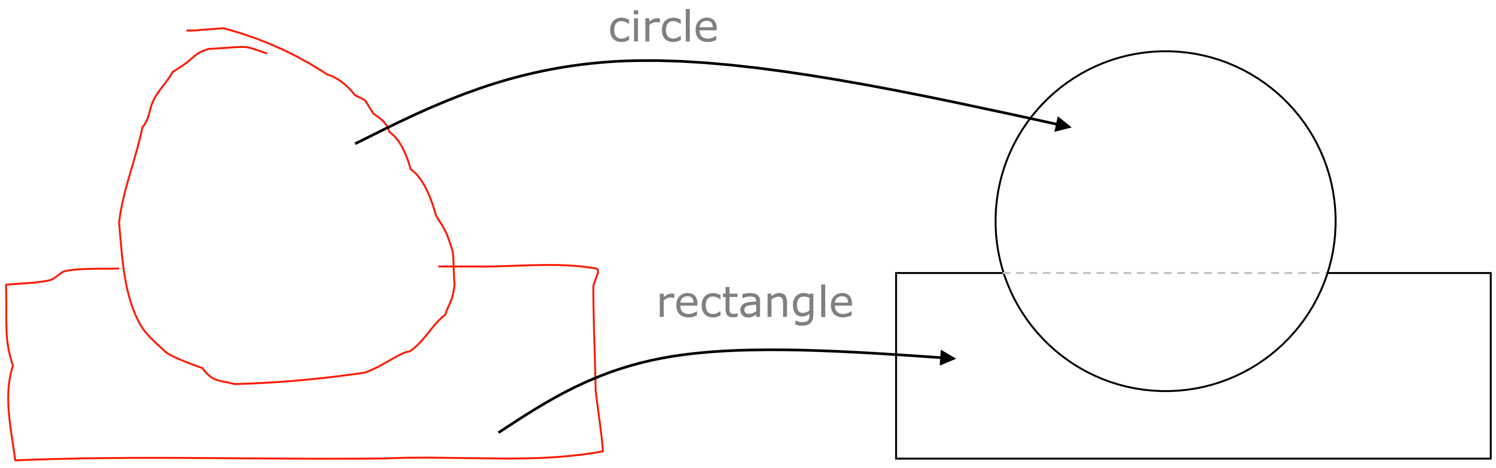


Locally connected

vs.

Globally separate

Stroke recognition



Stroke Processing

- *Filtering, neatening, beautification* can also be considered as stroke processing.
- Segmentation, classification, recognition.
- Regularization.
- Abstraction.
- Oversketching.
- Gestures.

Stroke grouping and regularization

Gestalt Principle

“The whole is greater than the sum of its parts”

Gestalt grouping and regularization

- Similarity
- Symmetry
- Continuation
- Closure
- Proximity

Regularization makes strokes that are nearly isometric, parallel, symmetric, perpendicular etc. precisely so!

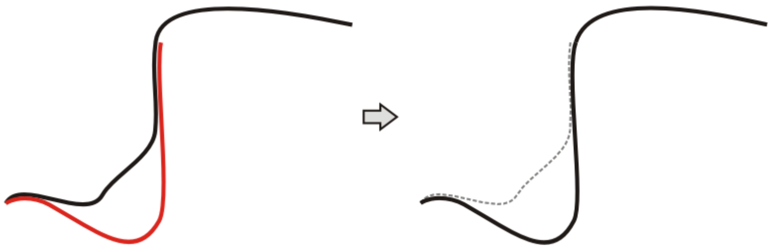
Stroke Abstraction

Stroke neatening that captures the essence of the stroke.

Stroke Oversketching

Interactive sketch correction

1. Find affected region
2. Splice in new stroke
3. Smooth connection



Gesture support

- Ad-hoc or pre-defined: “Recognizers that use heuristics specifically tuned to a pre-defined set of gestures.” [Wobbrock 2007]
  - Application specific: shorthand, chinese Brush Painting, musical scores, chemical formulas.
  - Platform specific: gesture libraries.

- Template-based or systematic.
  - Toolkit or framework
  - Simple algorithm

Ad-hoc vs. template-based

- Ad-hoc can recognize more complex gestures.
- Harder to train template-based gestures.
- Better consistency of gestural use in ad-hoc systems.
- Better gesture collision handling in ad-hoc systems.

- Ad-hoc doesn’t allow new gestures and limited customization.

GRANDMA approach

1. Encode gestures as a linear function of 13 features.

2. Draw a gesture ~15 times.

3. Train asset of feature weights for each gesture.

4. Classify gestures based on highest feature function score.

$1 recognizer

- Most recognizers are hard to write and involve a certain amount of machine learning.
- Toolkits are not available in every setting.

$1 goals

- Resilience to sampling.
- Require no advance math.
- Small code.
- Fast.
- 1-gesture training.
- Return an N-best list with scores.

$1 algorithm

- Resample the input
  - N evenly spaced points
- Rotate
  - “Indicative” angle between centroid and start point
- Scale
  - Reference square
- Re-rotate and Score
  - Score built from average distance between candidate and template points

Limitations

- Cannot distinguish between gestures whose identities depend on aspect ratios, orientations.
  - Square from rectangle
  - Up arrow from down arrow
- Cannot be distinguished based on speed.
- Only single strokes.
- Stroke order is important.
- Closed strokes?
- Gestalt gestures!

Take-aways

- Understand your application:
  - Does it need strokes?
  - Are strokes natural and of low-complexity, 2D or 3D?
- Source of stroke error?
- Only jump to 3D if you need to!
- Use stroke dynamics and temporal order carefully.
- Make reasonable assumptions.